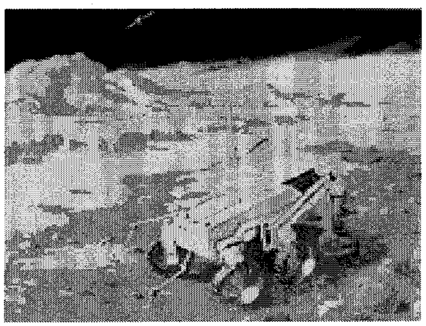
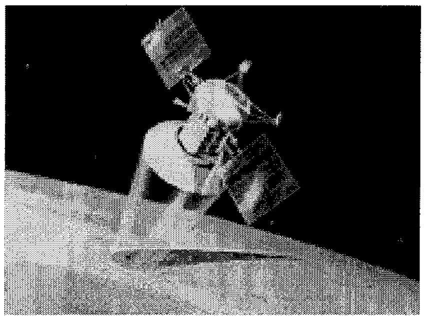
Development of a Spaceborne Embedded Cluster

Daniel S. Katz, Paul L. Springer

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California



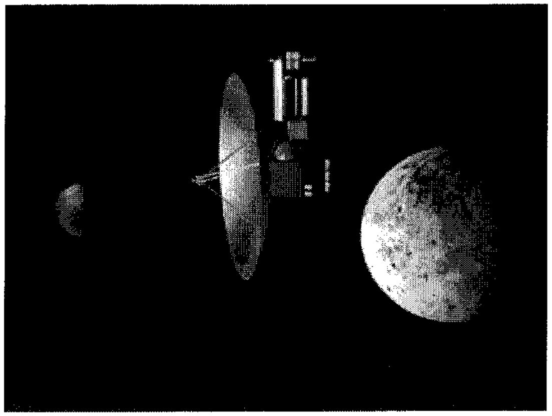
Autonomous Vehicles



High Data Rate Instruments

REE Vision

Move Earth-based Scalable Supercomputing Technology into Space



Background

- **Funded by Office of Space Science (Code S) as part of NASA's High Performance Computing and Communications Program**
- **Started in FY1996**

REE Impact on NASA and DOD Missions by FY05

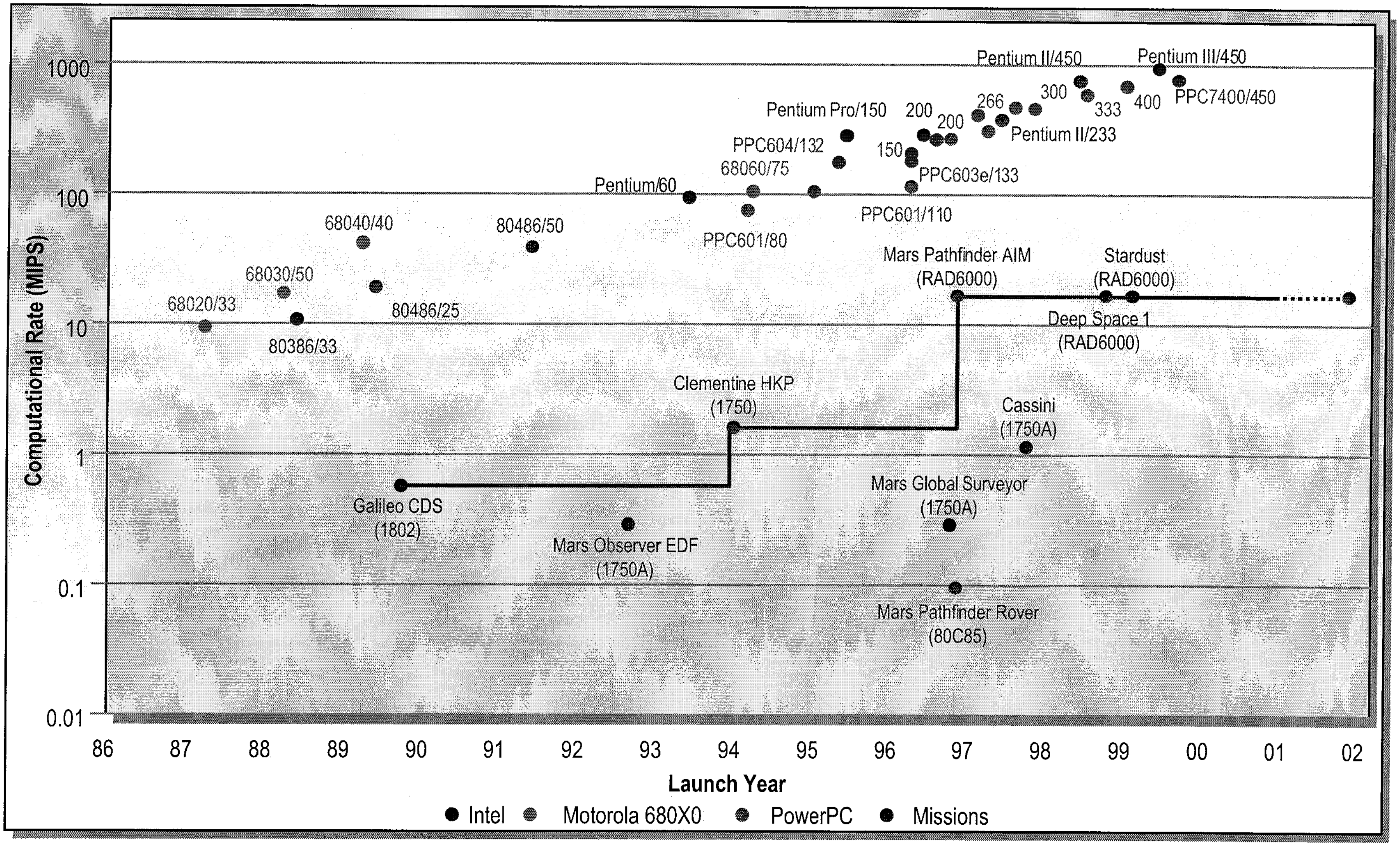
Faster - **Fly State-of-the-Art Commercial Computing Technologies within 18 months of availability on the ground**

Better - **Onboard computer operating at > 300MOPS/watt scalable to mission requirements *(> 100x Mars Pathfinder power performance)***

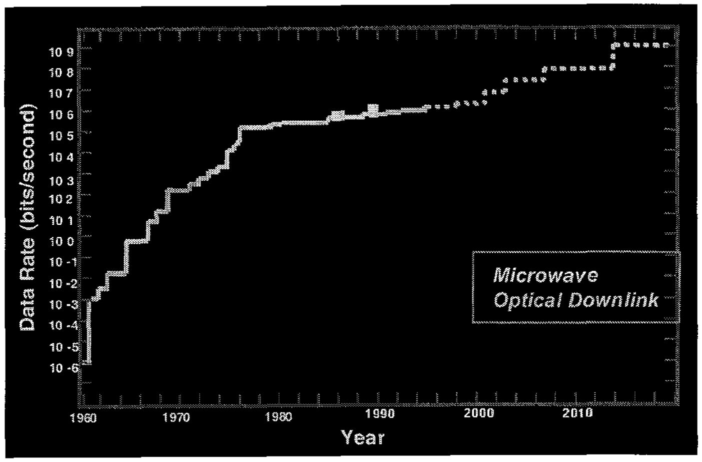
Cheaper - **No high cost radiation hardened processors or special purpose architectures**

Development of a Spaceborne Embedded Cluster

Space Flight & Microcomputer Processors



Bandwidth & Latency



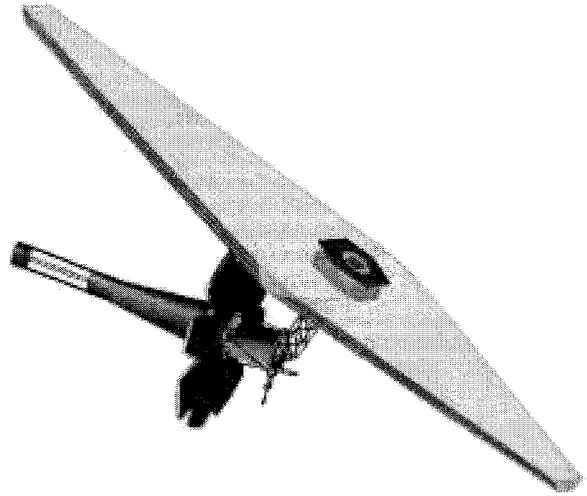
- **Bandwidth is relatively constant, compared with increasing ability of sensors to produce data**
- **Latency**
  - To Mars ranges from 3 minutes to 20 minutes one way
  - To L2 is about a minute one way
  - These times prohibit most automated response with ground-based computing in the loop

Science Application Teams

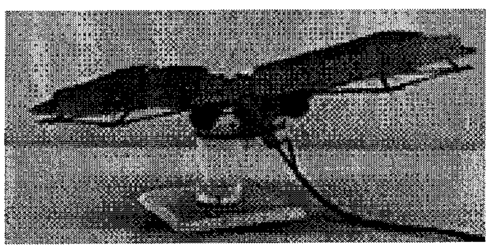
- Background
  - Enabling new and better science is a primary goal for REE
  - A new generation of Mission Scientists is emerging which sees the value of significant onboard computing capability
    - Mission Scientists still want the most data bits possible sent back to the ground
    - But bandwidth to the ground is stagnant, while instrument data rates continue to rise dramatically
    - Ground operations costs are a major component of mission costs
- Science Application Teams chosen to:
  - Represent the diversity of NASA onboard computing of the future
  - Drive architecture and system software requirements
  - Demonstrate the benefit of highly capable computing onboard
- Science Application Teams will:
  - Prototype applications based on their mission concepts
  - Port and demonstrate applications on the 1st Generation Testbed
  - Use their experiences with REE to influence some of their mission design decisions

Next Generation Space Telescope Team

REE Principle Investigator: Dr. John Mather, NGST Study Scientist



SCIENCE OBJECTIVES
- Study the birth of the first galaxies
- Determine the shape and fate of the universe
- Study formation of stars and planets
- Observe the chemical evolution of the universe
- Probe the nature of dark matter

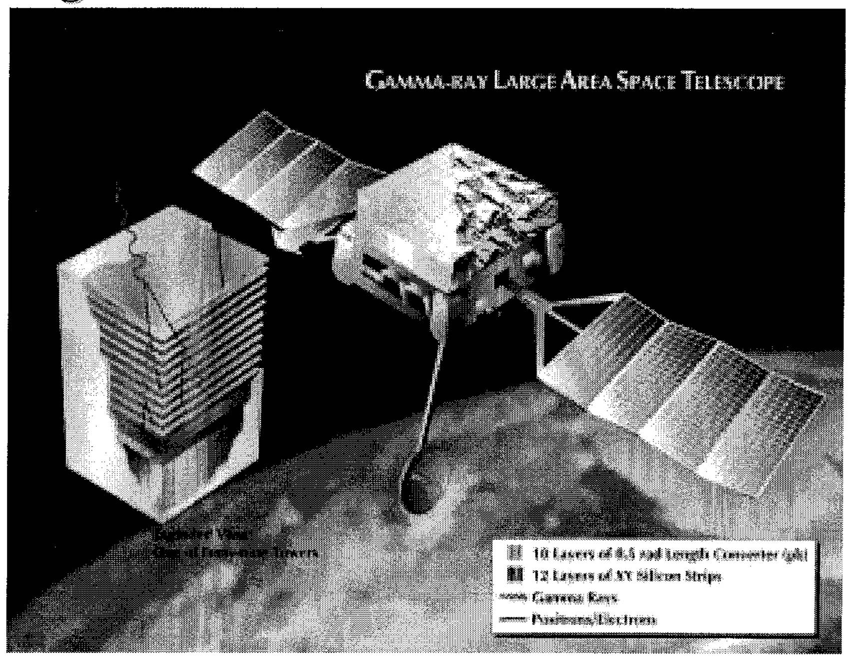


TECHNOLOGY HIGHLIGHTS
- Precision deployable and inflatable structures
- Large, low area density cold active optics
- Removing cosmic ray interactions from CCD readouts
- Simulation based design
- Passive cooling
- Autonomous operations and onboard scheduling

Gamma Ray Large Area Space Telescope

REE Principal Investigator: Professor Peter Michelson, Stanford University, GLAST Principle Investigator

- GLAST will probe active galactic nuclei (spectral shape and cutoff), study gamma-ray pulsars, respond in real-time to gamma-ray bursts.
- GLAST will produce 5-10 Megabytes per second after sparse readout, mapping into 50 MIPS of computing requirements to meet the requirements for the baseline mission.
- New science addressed by GLAST focuses on transient events of a few days in AGNs and .01–100 seconds in gamma-ray bursts.
- REE could enable GLAST to produce 10x this data volume if it were to do most of its background discrimination in software. This would allow real-time identification of gamma-ray bursts, and permit the mission scientists to extract secondary science from the "background."

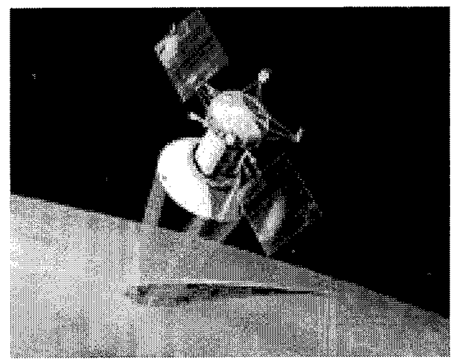


GLAST is a high-energy gamma-ray observatory designed for making observations of celestial sources in the range from 10 MeV to 300 GeV.

Orbiting Thermal Imaging Spectrometer

REE Principal Investigator - Alan Gillespie/U. Washington, Member of the ASTER Science Team

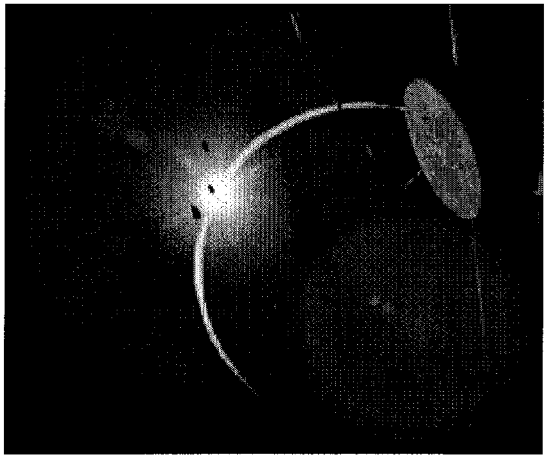
- Similar to Sacagawea:
  - Polar-orbiting high-resolution imaging infrared spectrometer (8-12 μm)
  - 64 bands of 12-bit data over a 21 swath at 30 m/pixel every 3.1 sec
  - Raw data rate of 30 MB/s
  - Designed to map emissivity of the Earth's surface to:
    - Map lithologic composition
    - Enable surface temperature recovery over all surfaces



- Onboard Processing
  - Characterize and compensate for atmospheric effects
  - Calculate land surface temperatures and emissivity spectra
  - Automatically convert the emissivity data to a thematic map

Solar Terrestrial Probe Program

REE Principal Investigator - Steve Curtis/GSFC STPP Study Scientist

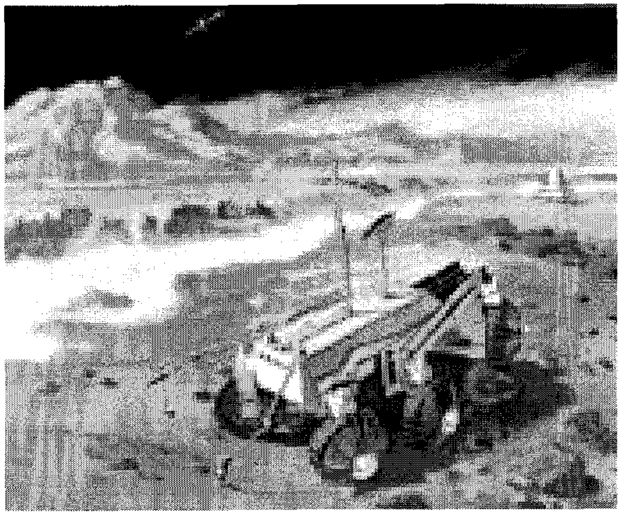


- **Solar Terrestrial Probe Goal**
  - Real-time quantitative understanding of the flow of energy,mass,momentum and radiation from the sun to the earth
    - Solar processes, flares and mass ejections
    - Interplanetary space and solar wind
    - Earth's magnetosphere and upper atmosphere
- **Mission Onboard Processing Applications - Data Reduction!**
  - Magnetospheric Constellation Mission
    - 50- 100 identical, spinning 10 kg spacecraft with on-board plasma analyzers (ions and electrons), a magnetometer and an electrometer
    - Compute moments of a sample plasma distribution function onboard
  - Low Frequency Radio Astronomy Imaging (ALFA/SIRA mission)
    - 16 - 64 formation flying spacecraft using interferometry to produce low frequency maps and two dimensional imaging of solar disturbances.
    - Compute pairs of time series (120+) to find the correlation maximum

Autonomous Mars Rover Science

REE Principal Investigator: R. Steve Saunders/JPL Mars '01 Lander PI

- Autonomous optimal terrain navigation
  - Stereo vision
  - Path planning from collected data
  - Autonomous determination of experiment schedule
  - Opportunistic scheduling



- Autonomous Field Geology
  - "Computational Geologist"
  - The rover returns analysis - not only data

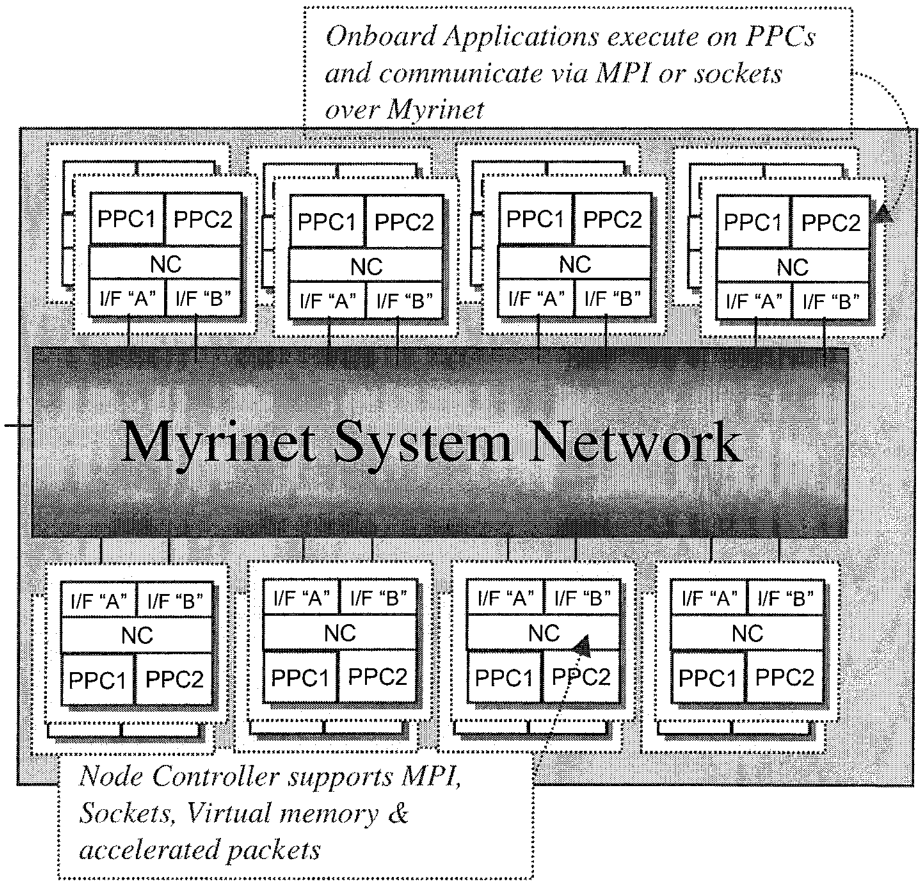
Radiation Environment for Applications

- Model Inputs
  - 3 orbit scenarios
    - Low Earth, 28° inclination
    - Geosynchronous, nominal solar activity
    - Geosynchronous, JPL "design case" solar flare, 100 mil aluminum shielding
  - All testbed components
  - Latch, gate fault capture rates based on preliminary analysis of PPC750 radiation testing
  - Assume memory and L2 cache are protected by EDAC
- Approximate predicted fault rates
  - Per Node (2 PCC750s, 1 Node Controller, 1 Network Switch)
  - Actual errors realized is lower since some faults have no effect
    - For one application tested, ~70% of faults cause no error

Orbit	Total Faults/Hr
LEO	~5
GEO, Nominal	~10
GEO, Flare	~100

REE First Generation Testbed Capabilities

- ~ 35 Million Operations (peak) per second per watt of power consumed
  - > 10x the power performance on Mars Pathfinder
  - Includes ALL component power (processors, memory, network)
- Communication between processors at 132 MB/s
- 128 MB EDAC memory per node
- No single point of failure
- Automatic reconfiguration around failed components
- Fault injection capability for every software accessible component
  - Processors, Memory, Network
  - Replicates radiation induced fault environment in the lab for experimentation & software validation
- COTS real time OS (Lynx)
- COTS programming environment, tools



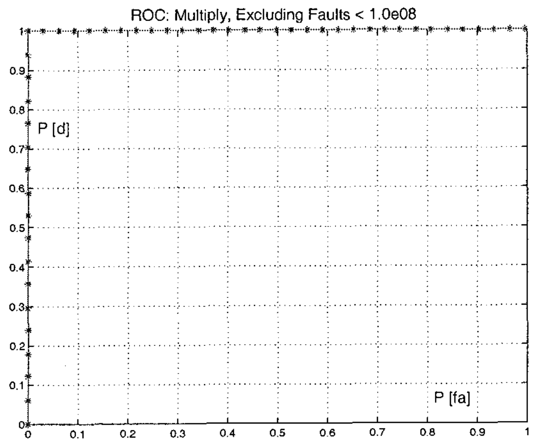
Faults and Errors

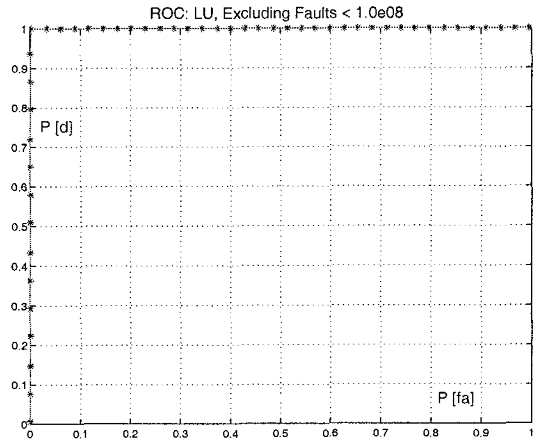
- Radiation environment causes faults
  - Most (>99.9%) of faults are transient, single event upsets (SEUs)
- Faults cause errors
  - Good Errors
    - Cause the node to crash
    - Cause the application to crash
    - Cause the application to hang
  - Bad Errors
    - Change application data
      - Application may complete, but the output may be wrong
- System Software can detect the good errors
  - Restarting the application/rollback/reboot is acceptable
- Applications must detect bad errors
  - Using Algorithm-Based Fault Tolerance (ABFT), assertion checking, other techniques

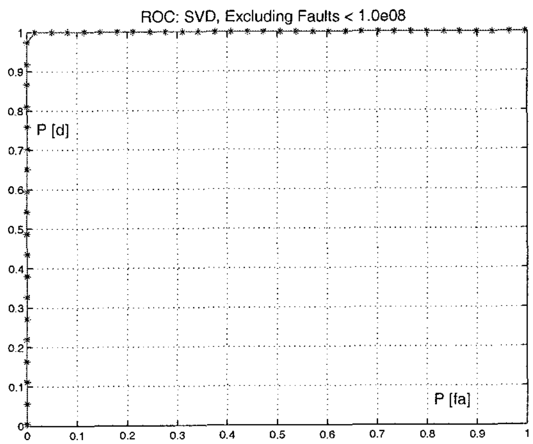
Algorithm-Based Fault Tolerance

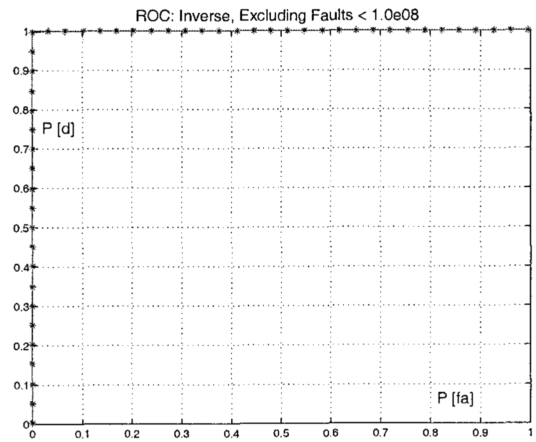
- Started in 1984 with Huang and Abraham
  - Initial motivation was systolic arrays
  - Abraham and his students continued to develop ABFT throughout 1980s
- Relationship to convolutional coding noticed
- Picked up in early 90s by a group of linear algebraists (Boley et al., Boley and Luk)
- ABFT techniques exist for many numerical algorithms
  - Matrix multiply, LU decomposition, QR decomposition, single value decomposition (SVD), fast Fourier transform (FFT)
  - Require an error tolerance
    - setting of this error tolerance involves a trade-off between missing errors and false positives
- ABFT can correct as well as detect errors
  - Currently, we are focusing on error detection, using result checking
    - If (transient) errors are detected, the routine is re-run

ABFT Results









Receiver Operating Characteristic (ROC) curves (fault-detection rate vs. false alarm rate) for random matrices of bounded condition number ($< 10^8$), excluding faults of relative size $< 10^{-8}$

ABFT Results (cont.)

- We have implemented a robust version of ScaLAPACK (on top of MPI) which detects errors using ABFT techniques
  - To the best of our knowledge, this is the first wrapping of a general purpose parallel library with an ABFT shell
  - Interface the same as standard ScaLAPACK with the addition of an extra error return code
  - For reasonable matrices, we can catch >99% (>97% for SVD) of significant errors with no false alarms
- ABFT version of FFTW recently completed
  - We can catch >98% of significant errors with no false alarms
- Testing to date has been algorithmic
- Intense fault-injection testing has just begun

REE Results-to-Date

- Scalable applications have been delivered and used
  - 9 proposed applications have been delivered to JPL
  - 7 are currently running on an embedded system
  - We have shown throughput increases of 18x - 62x over current radiation hardened processors (RAD 6000)
  - We have demonstrated good scalability and speed-up on our initial embedded testbed.
- ABFT-wrapped libraries have been developed for linear algebra, FFT
  - Routines have been rigorously tested
  - Next step is for the applications to use these libraries under fault injection experiments
- A number of questions still need to be answered...

Open Questions

- **What fault rates *and fault effects* will occur?**
  - The radiation environment is known; understanding effects of environment has just been started)
- **What percentage of faults can be detected without replication?**
  - Using ABFT and other techniques to check for incorrect answers
- **What is the overhead and coverage of AFT?**
  - Each technique (ABFT, signature checks, recovery blocks, etc.) should be tested to determine cost-benefit tradeoff
  - Heading towards offering a library of techniques to be chosen from my mission developers depending on reliability/power/timing tradeoffs
- **Is checkpointing/rollback sufficient to recover from faults?**
  - What's the cost-benefit tradeoff?
  - Can the state of REE applications be made sufficiently small that the overhead of checkpointing is not prohibitive?